update_mmu_cache is used to update TLB entries after do_pagefault. It used

```
       invalidate_itlb_mapping(addr);
       invalidate_dtlb_mapping(addr);
```

to invalidate TLB entry that appeared with the fault. This works with single CPU, but in SMP case update_mmu_cache may be called on a CPU different from that that took the exception. Without TLB sanity debugging this leads to looping over the instruction that caused initial pagefault until the wrong TLB entry is kicked out. With TLB sanity debugging enabled this BUGs right away.

The fix:

```
diff --git a/arch/xtensa/mm/cache.c b/arch/xtensa/mm/cache.c
index 81edeab..4a67bb7 100644
--- a/arch/xtensa/mm/cache.c
+++ b/arch/xtensa/mm/cache.c
@@ -159,8 +159,7 @@ update_mmu_cache(struct vm_area_struct * vma, unsigned
long addr, pte_t *ptep)

        /* Invalidate old entry in TLBs */

-       invalidate_itlb_mapping(addr);
-       invalidate_dtlb_mapping(addr);
+       flush_tlb_page(vma, addr);

 #if (DCACHE_WAY_SIZE > PAGE_SIZE) && XCHAL_DCACHE_IS_WRITEBACK
```